2011

A Threat to Cyber Resilience: A Malware Rebirthing Botnet

Murray Brand
Edith Cowan University

Craig Valli
Edith Cowan University

Andrew Woodward
Edith Cowan University

Originally published in the Proceedings of the 2nd International Cyber Resilience Conference, Edith Cowan University, Perth Western Australia, 1st - 2nd August 2011

A THREAT TO CYBER RESILIENCE: A MALWARE REBIRTHING BOTNET

M. Brand, C. Valli, A. Woodward
secau - Security Research Centre, School of Computer and Security Science,
Edith Cowan University, Perth, Western Australia

m.brand@.ecu.edu.au, c.valli@ecu.edu.au, a.woodward@ecu.edu.au

Abstract

This paper presents a threat to cyber resilience in the form of a conceptual model of a malware rebirthing botnet which can be used in a variety of scenarios. It can be used to collect existing malware and rebirth it with new functionality and signatures that will avoid detection by AV software and hinder analysis. The botnet can then use the customized malware to target an organization with an orchestrated attack from the member machines in the botnet for a variety of malicious purposes, including information warfare applications. Alternatively, it can also be used to inject known malware signatures into otherwise non malicious code and traffic to overloading the sensors and processing systems employed by intrusion detection and prevention systems to create a denial of confidence of the sensors and detection systems. This could be used as a force multiplier in asymmetric warfare applications to create confusion and distraction whilst attacks are made on other defensive fronts.

Keywords

malware, botnet, malware rebirthing, anti-analysis, information warfare, denial of confidence.

INTRODUCTION

This paper proposes a new model for a web robot network (botnet) that could be employed as an offensive, information warfare weapon in a variety of ways. This model, referred to as a malicious software (Malware) Rebirthing Botnet (MRB), differs in functionality to known botnets by first acting as a honeypot to collect malware. The collected malware can then be modified using a technique coined by the authors as 'malware rebirthing', such that the original virus signature of the malware is significantly altered in a rebirthing suite. The rebirthing suite modifies the original functionality, adds new functionality and inserts analysis avoidance techniques. The rebirthed malware could then be unleashed by the member machines of the botnet, at specified targets, in a controlled manner, under the direction of a Command and Control (C&C) infrastructure. Machines in targeted networks could then be infected by customized malware which is unlikely to be recognized by Anti-Virus (AV) software because signatures will very likely not exist for the rebirthed malware if it is used in directed and customized attacks (Brand, Valli, & Woodward, 2010b).

Recognition of malware is dependent upon an analyst having already analyzed the behaviour of the malware and extracted an identifying signature (Harbour, 2007). Malware typically employs anti-analysis techniques to try to avoid detection and to hinder analysis techniques (Brand, 2010). The rebirthed malware could also incorporate a range of anti-analysis techniques in a similar manner. Once on a target network, the malware can proceed to take control of the machines on the network and perform any number of malicious actions. This could include, but is not limited to, taking over control of any attached critical infrastructure, denying the use of the infected machines themselves, destruction of data or flooding the internet with new malware such that existing, popular paradigms for malware detection and removal are rendered ineffectual.

An alternative use of the MRB could be to insert known malware signatures into otherwise non malicious code and network traffic to overload the sensors and processing systems of intrusion detection and prevention systems. This could be used to create confusion and create a diversion whilst attacks are conducted on other defensive fronts. The principle of a denial of confidence can also be used in an attack where all, or as much as possible, installed software on targeted systems is salted with known malware signatures. Malware signatures used by detection systems are fairly short and can be inserted quite easily into existing programs and files. This principle of attack could be referred to as "salting the earth", with the objective of denying confidence in the integrity of systems. At the very least, it would tie up resources including incident responders, digital forensic investigators, system and network administrators and potentially deny use of the system to operators and infrastructure services.

BOTNETS

Botnets are very large collections of computers that are under the authority of a C&C infrastructure that are employed for a variety of nefarious purposes including, but not limited to, identity theft, spam e-mail campaigns, Distributed Denial of Service (DDoS) attacks and running spyware. DDoS in particular are events in which the members of the botnet collectively orchestrate an attack against a specific target directed by the C&C to flood the internet connection of the target such that the targeted network cannot send and receive legitimate internet traffic. Ordinarily such an attack is stopped at outer perimeter defences such as firewalls, or detected by internal sensors used by intrusion detection and prevention systems. Botnets are generally under the control of criminal organizations, but the suggestion for their use by military organizations is not unprecedented (Williamson, 2008).

MALWARE COLLECTION

Malware can be collected by software that emulates known software vulnerabilities, which can then download network based malware such as worms that attempt to exploit the vulnerabilities. Malware researchers set up malware collection machines as sensors that collect malware that can be used for research purposes and by AV software companies for the extraction of malware signatures and behaviour monitoring. The source code of such malware collection software is freely available and is well understood (Nepenthes, 2006). Existing honeypot, open source software can then be modified and extended by the use of modules that emulate known vulnerabilities.

ANTI-ANALYSIS TECHNIQUES

A plethora of anti-analysis techniques can be employed by malware in an attempt to hinder the malware analyst and to avoid detection by signature based AV software (Brand, Valli, & Woodward, 2010a). These techniques can include run-time packers, protectors, rootkits, anti-memory, anti-disassembler, anti-debuggers, anti-emulation, anti-analysis, anti-tools, anti-process, anti-online analysis and anti-hardware (Falliere, 2007; Ferrie, 2008; Yason, 2007). Run-time packers obfuscate and compress code which has to be run or emulated to read the original instructions. This means that it is very difficult to determine the nature of the code without running it, and the signature of the code can be easily altered by using different packers. Thousands of known software packers exist and customized packers can be written using freely available code as a reference. However, running the malware to unpack it, gives control to the malware process and it can use techniques to determine if it is being analyzed. If it determines that it is being analyzed, it can then use deceptive techniques to avoid showing the analyst its real intentions.

Protectors tend to make the original code much larger in size and add features such as encryption and other techniques that make the code much harder to analyse than code that has been packed by a run-time packer. This can include the use of anti-memory techniques that make it more difficult to analyse code that has been allowed to run, but dumped from memory for analysis. If the code has been managed to be unpacked in memory and dumped, anti-disassemblers techniques can be used to produce false disassemblies that hinder the analyst. Anti-debugger techniques are used by malware to detect the presence of a debugger. A debugger is one of the most useful tools to the analyst. Anti-debugging techniques make it more difficult for the analyst to step through, or run the code. Anti-emulators exist to detect the presence of virtual environments, which is a common test environment for analysts to analyse the malware in. Online analysis engines exist to analyse malware that is submitted for analysis, but the presence of these can also be detected. The use of popular analysis tools can also be detected by the malware whilst it is being run.

Although a high degree of skill is required to analyse malware (Valli & Brand, 2008), the use of anti-analysis techniques can be detected and mitigated (Brand, 2009). Essentially this can be achieved through the use of scripting languages and can be automated (Eagle, 2008; Erdélyi, 2008).

FAILURE OF AV SOFTWARE

AV software typically uses a combination of signature matching, heuristics and file integrity checking to detect malware (Farwell, 2004). Signature matching relies on the recognition of a signature that has been previously extracted from analyzed malware. If the malware has not been previously analyzed and had a signature extracted and updated on client computers, signature matching is likely to fail. Heuristics flag any behaviour that is outside the normal operating parameters of the system and is prone to high levels of false positives. Integrity checking relies on detecting changes to known files and is reliant upon having a 100% clean system in the first place and is only effective after an instance of malware has been detected. This could be too late because the malware will most likely have disabled security including AV software. AV software that relies on these techniques is recognized by AV researchers to be less than optimal (Farwell, 2004; Mila Dalla, Mihai, Somesh,

& Saumya, 2008; Szewczyk & Brand, 2008; W. Yan, Zhang, & Ansari, 2008; Z. Yan & Inge, 2008; Zhou & Meador Inge, 2008).

In May 2009, the number of known malware signatures is approaching 3.75 million signatures (Triumfant, 2009). This emphasizes the difficulty in updating virus signature databases for computers. Given the range of anti-analysis techniques discussed in the previous section, it is not difficult to conceive that AV software vendors are well behind in the malware arms race. The malware rebirthing bot takes advantage of these conditions.

MALWARE REBIRTHING SUITE MODEL

The proposed model presents the concept of a malware rebirthing suite that takes collected malware as an input and produces a new version of malware with altered original functionality and added new functionality. The purpose of this is to customize the malware to attack the target, avoid detection by AV software and hinder analysis by malware analysts.

It is possible to evade AV software detection by altering a few bytes of the malware that are in common with the signature used by the AV software. However, measures of self similarity including phylogenic techniques can be used to detect malware where small changes have been made to the code (Carrera & Halvar Flake, 2008; Karim, Walenstein, Lakhotia, & Parida, 2005).

A functional block diagram of the proposed malware rebirthing suite component is depicted in Figure 1. Under the direction of the management component of the bot, malware collected from the honeypot is rebirthed. The rebirthing process combines anti analysis techniques, modules of new or customized malicious functionality to the original malware. After merging the modules, the result is further transformed by a customized packer or protector.

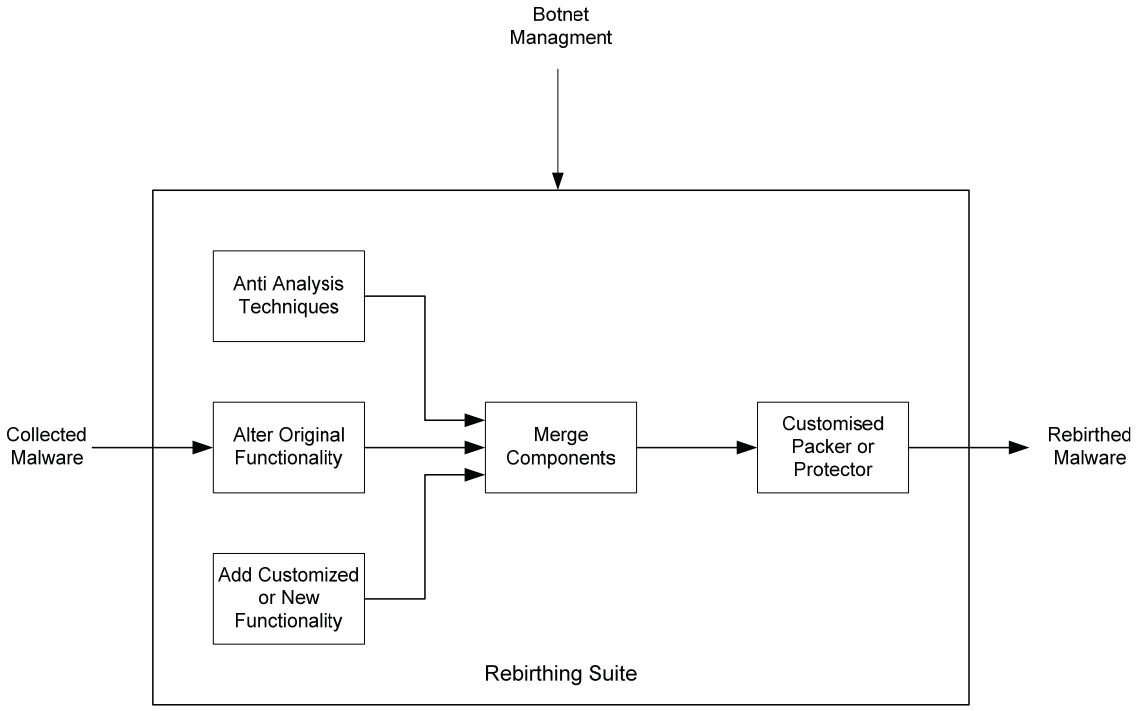

Figure 1: Rebirthing Suite Functional Block Diagram

CONCEPT OF OPERATION

A very simple concept of operation of the malware rebirthing bot is outlined in the following, high level, sequence of events.

1. Release worms on the internet that install the malware rebirthing bot on victim machines for a variety of target platforms.
2. The installed bot renders security measures such as AV software and Intrusion Detection Systems (IDS) on victim machine ineffectual.

3. Infected machines communicate successful infection to C&C and download updated releases of malware collection software and the malware rebirthing suite, appropriate for the victim platform, as required.
4. Infected machines continue to listen for commands from the C&C and to report status.
5. The installed bot converts the infected machine into a honeypot and begins to collect malware.
6. Collected malware is rebirthed and distributed amongst members of the botnet.
7. At the command of the C&C, an exploit module, appropriate for the target of interest, is added to the rebirthed malware that is suitable to be used to attack the target of interest is launched.

Figure 2 provides a conceptual model of the malware rebirthing bot once it is installed on a target machine. It shows that the bot emulates known vulnerabilities which are exploited by network based malicious software such as worms and other bots. The payload of the malware is captured and saved under the direction of a management process. The management process communicates with a C&C which directs the activities of the bot and provides updates to modules. Bot management directs how the rebirthing suite operates including developing permutations of the malware appropriate for specific types of attacks. On reception of an instruction to attack a particular target, the management process attaches an appropriate exploit module suitable for the target and engagement commences.

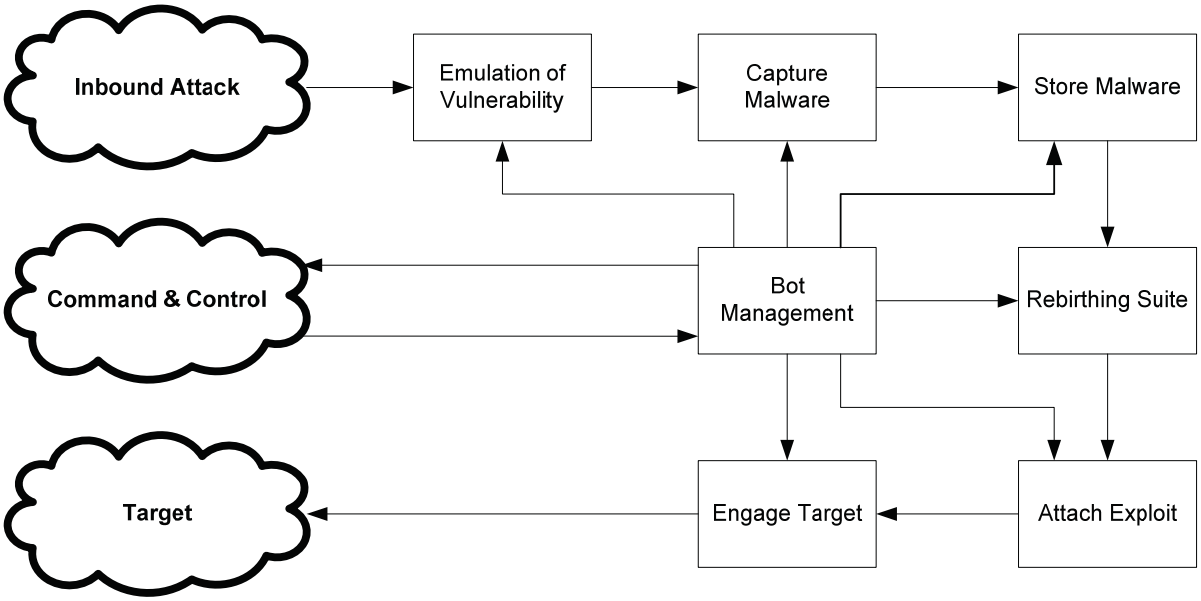

Figure 2: Conceptual Model of Functioning Malware Rebirthing Bot

Ideally, the malware rebirthing botnet would alter the original functionality of the collected malware and add new functionality to engage the target network computers in a customized manner. Conceivably, it need not be considerably sophisticated to be effective enough to achieve this. At its most simple level, the malware rebirthing botnet could flood a target network with a variety of customized malware that does not have a coordinated objective. The result of such an attack would most likely be the requirement to take all machines off the network, simultaneously, to have their original software reinstalled because disinfection information will not exist, and infected computers will continue to infect other computers. At the other end of the spectrum, customized malicious software that does have a coordinated objective could be used to take over control of critical infrastructure or network operations in a very stealthy manner. In a worst case scenario, the malware rebirthing botnet could be used as a doomsday weapon on the internet. In such a case, its only purpose could be to continually generate new worms from collected and rebirthed malware to indiscriminately flood the internet with worms and bots such that the combined weight of AV detection software companies cannot keep up with the generation of new malware.

MITIGATION STRATEGIES

Given that the performance of AV software is considered to be less than ideal, alternative techniques for malware detection need to be considered. An alternative technique worthy of consideration is detection of software that actively employs anti-analysis techniques. In general, the Intellectual Property (IP) of software can be protected from simple reverse engineering by incorporating anti-analysis techniques. However, in a study conducted by Brand (2009), 98.9% of 829 collected malware specimens employed simple anti-analysis techniques. Detection of the use of anti-analysis techniques could be used as an initial detector of malware. A

possible mitigation strategy could be to scan computers for collections of software that exhibit such behaviour. Detection of collections of such files can then be used to trigger remedial action. Popular AV software does recognize software that employs packers and protectors and flags such discoveries as suspicious. However, as noted above, the range of anti-forensic techniques is extensive and is not limited to packers and protectors. Additionally, malicious software can and will, disable AV software.

Another mitigation strategy could be to employ a technique referred to as 'whitelisting'. The objective of whitelisting is only to run software on a network from a pre-approved list (Kotadia & Winterford, 2008). This is in contrast to the normal practice of 'blacklisting', where computers are prohibited from running software that is on the list. Although not a widely adopted technique, whitelisting would effectively reject malware since it will not be on the pre-approved list. Of course, this technique would be rendered useless if the pre-approved list itself was compromised in a pre-emptive strike.

CONCLUDING REMARKS

This paper has presented a model of a Malware Rebirthing Botnet as a threat to Cyber Resilience. It can be used in a number of ways, and not limited to, collection and modification of existing malware to attack systems, insertion of known malware signatures into non malicious code and network traffic to overload sensors and processing systems to achieve a denial of confidence. It can be used to salt good programs with short, known malware signatures to trigger malware detection systems such that the systems are no longer trustworthy and taken off line for detailed attention.

Realistically, the malware rebirthing botnet could be implemented and would be a devastating information warfare weapon. All of the components of the malware rebirthing botnet exist, except for the malware rebirthing suite. Although the rebirthing suite has not been fully implemented at this point in time, the tools and techniques exist to develop it and it is not considered by the authors to be overly technically difficult to implement. There is little doubt that if the malware rebirthing botnet was unleashed, a new paradigm for malware detection would be required and would supersede existing AV software detection techniques such as signature recognition, heuristics and file integrity checking.

REFERENCES

Brand, M. (2009). Analysis Avoidance Techniques of Malicious Software. Unpublished Thesis. Edith Cowan University, Perth, WA.

Brand, M. (2010). Analysis Avoidance Techniques of Malicious Software: Edith Cowan University, Perth, WA.

Brand, M., Valli, C., & Woodward, A. (2010a). *Lessons Learned from an Investigation into the Analysis Avoidance Techniques of Malicious Software.* Paper presented at the 8th Australian Digital Forensics Conference, Perth, WA.

Brand, M., Valli, C., & Woodward, A. (2010b). Malware Forensics: Discovery of the intent of Deception. *Journal of Digital Forensics, Security and Law, 5*(4), 31-41.

Carrera, E., & Halvar Flake. (2008). *Automated Structural Classification of Malware.* Paper presented at the SOURCE Boston 2008, Boston.

Eagle, C. (2008). *The IDA Book*: No Starch Press.

Erdélyi, G. (2008). IDA Python.

Falliere, N. (2007). Windows Anti-Debug Reference. Retrieved October 1, 2007 from http://www.securityfocus.com/infocus/1893

Farwell, J. (2004). The Heart of the Matter, What Makes Antivirus Software Tick? Retrieved March 17, 2007, from http://www.smartcomputing.com/editorial/article.asp?article=articles/2004/s1511/24s11/24s11.asp&articleid=23737&guid=

Ferrie, P. (2008). *Anti-Unpacker Tricks*. Paper presented at the 2nd International Caro Workshop. from http://www.datasecurity-event.com/uploads/unpackers.pdf

Harbour, N. (2007). Stealth Secrets of the Malware Ninjas. Retrieved October 20, 2007 from https://www.blackhat.com/presentations/bh-usa-07/Harbour/Presentation/bh-usa-07-harbour.pdf

Karim, E., Walenstein, A., Lakhotia, A., & Parida, L. (2005). Malware Phylogeny Generation Using Permutations of Code. *Journal in Computer Virology, Volume 1, Numbers 1-2*, 13-23.

Kotadia, M., & Winterford, B. (2008). Is whitelisting the new blacklisting? , from http://www.zdnet.com.au/news/security/soa/Is-whitelisting-the-new-blacklisting-/0,130061744,339289355,00.htm?feed=pt_auscert

Mila Dalla, P., Mihai, C., Somesh, J., & Saumya, D. (2008). A semantics-based approach to malware detection. *ACM Trans. Program. Lang. Syst., 30*(5), 1-54.

Nepenthes. (2006). Nepenthes – Finest Collection. Retrieved March 16, 2004 from http://nepenthes.mwcollect.org

Szewczyk, P., & Brand, M. (2008). *Malware Detection and Removal: An Examination of Personal Anti-Virus Software.* Paper presented at the 6th Australian Digital Forensics Conference, Edith Cowan University, Mount Lawley Campus, Western Australia.

Triumfant. (2009). The Worldwide Malware Signature Counter. Retrieved 25 May 2009, from http://www.triumfant.com/Signature_Counter.asp

Valli, C., & Brand, M. (2008). *Malware Analysis Body of Knowledge.* Paper presented at the 6th Australian Digital Forensics Conference, Edith Cowan University, Mount Lawley Campus, Western Australia.

Williamson, C. (2008). Carpet bombing in cyberspace. Retrieved from http://www.armedforcesjournal.com/2008/05/3375884

Yan, W., Zhang, Z., & Ansari, N. (2008). Revealing Packed Malware. *IEEE Security and Privacy 6 (5)*, 65-69.

Yan, Z., & Inge, W. M. (2008). *Malware detection using adaptive data compression*. Paper presented at the Proceedings of the 1st ACM workshop on Workshop on AISec.

Yason, M. (2007). The Art of Unpacking. Retrieved Feb 12, 2008 from https://www.blackhat.com/presentations/bh-usa-07/Yason/Whitepaper/bh-usa-07-yason-WP.pdf

Zhou, Y., & Meador Inge, W. (2008). *Malware detection using adaptive data compression*. Paper presented at the Proceedings of the 1st ACM workshop on Workshop on AISec